package com.twitter.search.common.relevance.features;

import java.util.Map;

import java.util.Set;

import javax.annotation.Nullable;

import com.google.common.base.Preconditions;

import com.google.common.collect.ImmutableMap;

import com.google.common.collect.ImmutableSet;

import com.twitter.search.common.encoding.features.IntNormalizer;

import com.twitter.search.common.schema.earlybird.EarlybirdFieldConstants;

import static com.twitter.search.common.relevance.features.IntNormalizers.BOOLEAN\_NORMALIZER;

import static com.twitter.search.common.relevance.features.IntNormalizers.LEGACY\_NORMALIZER;

import static com.twitter.search.common.relevance.features.IntNormalizers.PARUS\_SCORE\_NORMALIZER;

import static com.twitter.search.common.relevance.features.IntNormalizers.SMART\_INTEGER\_NORMALIZER;

import static com.twitter.search.common.relevance.features.IntNormalizers.TIMESTAMP\_SEC\_TO\_HR\_NORMALIZER;

import static com.twitter.search.common.schema.earlybird.EarlybirdFieldConstants.EarlybirdFieldConstant;

/\*\*

\* An enum to represent all dynamic/realtime feature types we can update in the Signal Ingester.

\* It provides information for their normalization and their corresponding earlybird feature fields

\* and provides utils both producer (Signal Ingester) and consumer (Earlybird) side.

\*

\*/

public enum TweetFeatureType {

RETWEET (true, 0, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.RETWEET\_COUNT),

REPLY (true, 1, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.REPLY\_COUNT),

FAVORITE (true, 4, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.FAVORITE\_COUNT),

PARUS\_SCORE (false, 3, PARUS\_SCORE\_NORMALIZER,

EarlybirdFieldConstant.PARUS\_SCORE),

EMBEDS\_IMP\_COUNT (true, 10, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.EMBEDS\_IMPRESSION\_COUNT),

EMBEDS\_URL\_COUNT (true, 11, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.EMBEDS\_URL\_COUNT),

VIDEO\_VIEW (false, 12, LEGACY\_NORMALIZER,

EarlybirdFieldConstant.VIDEO\_VIEW\_COUNT),

// v2 engagement counters, they will eventually replace v1 counters above

RETWEET\_V2 (true, 13, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.RETWEET\_COUNT\_V2),

REPLY\_V2 (true, 14, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.REPLY\_COUNT\_V2),

FAVORITE\_V2 (true, 15, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.FAVORITE\_COUNT\_V2),

EMBEDS\_IMP\_COUNT\_V2 (true, 16, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.EMBEDS\_IMPRESSION\_COUNT\_V2),

EMBEDS\_URL\_COUNT\_V2 (true, 17, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.EMBEDS\_URL\_COUNT\_V2),

VIDEO\_VIEW\_V2 (false, 18, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.VIDEO\_VIEW\_COUNT\_V2),

// other new items

QUOTE (true, 19, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.QUOTE\_COUNT),

// weighted engagement counters

WEIGHTED\_RETWEET (true, 20, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.WEIGHTED\_RETWEET\_COUNT),

WEIGHTED\_REPLY (true, 21, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.WEIGHTED\_REPLY\_COUNT),

WEIGHTED\_FAVORITE (true, 22, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.WEIGHTED\_FAVORITE\_COUNT),

WEIGHTED\_QUOTE (true, 23, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.WEIGHTED\_QUOTE\_COUNT),

// tweet-level safety labels

LABEL\_ABUSIVE (false, 24, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_ABUSIVE\_FLAG),

LABEL\_ABUSIVE\_HI\_RCL (false, 25, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_ABUSIVE\_HI\_RCL\_FLAG),

LABEL\_DUP\_CONTENT (false, 26, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_DUP\_CONTENT\_FLAG),

LABEL\_NSFW\_HI\_PRC (false, 27, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_NSFW\_HI\_PRC\_FLAG),

LABEL\_NSFW\_HI\_RCL (false, 28, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_NSFW\_HI\_RCL\_FLAG),

LABEL\_SPAM (false, 29, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_SPAM\_FLAG),

LABEL\_SPAM\_HI\_RCL (false, 30, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.LABEL\_SPAM\_HI\_RCL\_FLAG),

PERISCOPE\_EXISTS (false, 32, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.PERISCOPE\_EXISTS),

PERISCOPE\_HAS\_BEEN\_FEATURED (false, 33, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.PERISCOPE\_HAS\_BEEN\_FEATURED),

PERISCOPE\_IS\_CURRENTLY\_FEATURED (false, 34, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.PERISCOPE\_IS\_CURRENTLY\_FEATURED),

PERISCOPE\_IS\_FROM\_QUALITY\_SOURCE(false, 35, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.PERISCOPE\_IS\_FROM\_QUALITY\_SOURCE),

PERISCOPE\_IS\_LIVE (false, 36, BOOLEAN\_NORMALIZER,

EarlybirdFieldConstant.PERISCOPE\_IS\_LIVE),

// decayed engagement counters

DECAYED\_RETWEET (true, 37, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.DECAYED\_RETWEET\_COUNT),

DECAYED\_REPLY (true, 38, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.DECAYED\_REPLY\_COUNT),

DECAYED\_FAVORITE (true, 39, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.DECAYED\_FAVORITE\_COUNT),

DECAYED\_QUOTE (true, 40, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.DECAYED\_QUOTE\_COUNT),

// timestamp of last engagement types

LAST\_RETWEET\_SINCE\_CREATION\_HR (false, 41, TIMESTAMP\_SEC\_TO\_HR\_NORMALIZER,

EarlybirdFieldConstant.LAST\_RETWEET\_SINCE\_CREATION\_HRS),

LAST\_REPLY\_SINCE\_CREATION\_HR (false, 42, TIMESTAMP\_SEC\_TO\_HR\_NORMALIZER,

EarlybirdFieldConstant.LAST\_REPLY\_SINCE\_CREATION\_HRS),

LAST\_FAVORITE\_SINCE\_CREATION\_HR (false, 43, TIMESTAMP\_SEC\_TO\_HR\_NORMALIZER,

EarlybirdFieldConstant.LAST\_FAVORITE\_SINCE\_CREATION\_HRS),

LAST\_QUOTE\_SINCE\_CREATION\_HR (false, 44, TIMESTAMP\_SEC\_TO\_HR\_NORMALIZER,

EarlybirdFieldConstant.LAST\_QUOTE\_SINCE\_CREATION\_HRS),

// fake engagement counters

FAKE\_RETWEET (true, 45, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.FAKE\_RETWEET\_COUNT),

FAKE\_REPLY (true, 46, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.FAKE\_REPLY\_COUNT),

FAKE\_FAVORITE (true, 47, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.FAKE\_FAVORITE\_COUNT),

FAKE\_QUOTE (true, 48, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.FAKE\_QUOTE\_COUNT),

// blink engagement counters

BLINK\_RETWEET (true, 49, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.BLINK\_RETWEET\_COUNT),

BLINK\_REPLY (true, 50, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.BLINK\_REPLY\_COUNT),

BLINK\_FAVORITE (true, 51, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.BLINK\_FAVORITE\_COUNT),

BLINK\_QUOTE (true, 52, SMART\_INTEGER\_NORMALIZER,

EarlybirdFieldConstant.BLINK\_QUOTE\_COUNT),

/\* semicolon in a single line to avoid polluting git blame \*/;

private static final Map<TweetFeatureType, TweetFeatureType> V2\_COUNTER\_MAP =

ImmutableMap.<TweetFeatureType, TweetFeatureType>builder()

.put(RETWEET, RETWEET\_V2)

.put(REPLY, REPLY\_V2)

.put(FAVORITE, FAVORITE\_V2)

.put(EMBEDS\_IMP\_COUNT, EMBEDS\_IMP\_COUNT\_V2)

.put(EMBEDS\_URL\_COUNT, EMBEDS\_URL\_COUNT\_V2)

.put(VIDEO\_VIEW, VIDEO\_VIEW\_V2)

.build();

private static final Map<TweetFeatureType, TweetFeatureType> WEIGHTED\_COUNTER\_MAP =

ImmutableMap.<TweetFeatureType, TweetFeatureType>builder()

.put(RETWEET, WEIGHTED\_RETWEET)

.put(REPLY, WEIGHTED\_REPLY)

.put(FAVORITE, WEIGHTED\_FAVORITE)

.put(QUOTE, WEIGHTED\_QUOTE)

.build();

private static final Map<TweetFeatureType, TweetFeatureType> DECAYED\_COUNTER\_MAP =

ImmutableMap.<TweetFeatureType, TweetFeatureType>builder()

.put(RETWEET, DECAYED\_RETWEET)

.put(REPLY, DECAYED\_REPLY)

.put(FAVORITE, DECAYED\_FAVORITE)

.put(QUOTE, DECAYED\_QUOTE)

.build();

private static final Map<TweetFeatureType, TweetFeatureType> DECAYED\_COUNTER\_TO\_ELAPSED\_TIME =

ImmutableMap.<TweetFeatureType, TweetFeatureType>builder()

.put(DECAYED\_RETWEET, LAST\_RETWEET\_SINCE\_CREATION\_HR)

.put(DECAYED\_REPLY, LAST\_REPLY\_SINCE\_CREATION\_HR)

.put(DECAYED\_FAVORITE, LAST\_FAVORITE\_SINCE\_CREATION\_HR)

.put(DECAYED\_QUOTE, LAST\_QUOTE\_SINCE\_CREATION\_HR)

.build();

private static final Set<TweetFeatureType> DECAYED\_FEATURES =

ImmutableSet.of(DECAYED\_RETWEET, DECAYED\_REPLY, DECAYED\_FAVORITE, DECAYED\_QUOTE);

private static final Set<TweetFeatureType> FAKE\_ENGAGEMENT\_FEATURES =

ImmutableSet.of(FAKE\_RETWEET, FAKE\_REPLY, FAKE\_FAVORITE, FAKE\_QUOTE);

private static final Set<TweetFeatureType> BLINK\_ENGAGEMENT\_FEATURES =

ImmutableSet.of(BLINK\_RETWEET, BLINK\_REPLY, BLINK\_FAVORITE, BLINK\_QUOTE);

@Nullable

public TweetFeatureType getV2Type() {

return V2\_COUNTER\_MAP.get(this);

}

@Nullable

public static TweetFeatureType getWeightedType(TweetFeatureType type) {

return WEIGHTED\_COUNTER\_MAP.get(type);

}

@Nullable

public static TweetFeatureType getDecayedType(TweetFeatureType type) {

return DECAYED\_COUNTER\_MAP.get(type);

}

// Whether this feature is incremental or direct value.

private final boolean incremental;

// This normalizer is used to (1) normalize the output value in DLIndexEventOutputBolt,

// (2) check value change.

private final IntNormalizer normalizer;

// value for composing cache key. It has to be unique and in increasing order.

private final int typeInt;

private final EarlybirdFieldConstants.EarlybirdFieldConstant earlybirdField;

private final IncrementChecker incrementChecker;

/\*\*

\* Constructing an enum for a type. The earlybirdField can be null if it's not prepared, they

\* can be here as placeholders but they can't be outputted.

\* The normalizer is null for the timestamp features that do not require normalization

\*/

TweetFeatureType(boolean incremental,

int typeInt,

IntNormalizer normalizer,

@Nullable EarlybirdFieldConstant earlybirdField) {

this.incremental = incremental;

this.typeInt = typeInt;

this.normalizer = normalizer;

this.earlybirdField = earlybirdField;

this.incrementChecker = new IncrementChecker(this);

}

public boolean isIncremental() {

return incremental;

}

public IntNormalizer getNormalizer() {

return normalizer;

}

public int getTypeInt() {

return typeInt;

}

public int normalize(double value) {

return normalizer.normalize(value);

}

public IncrementChecker getIncrementChecker() {

return incrementChecker;

}

public EarlybirdFieldConstant getEarlybirdField() {

return Preconditions.checkNotNull(earlybirdField);

}

public boolean hasEarlybirdField() {

return earlybirdField != null;

}

public boolean isDecayed() {

return DECAYED\_FEATURES.contains(this);

}

@Nullable

public TweetFeatureType getElapsedTimeFeatureType() {

return DECAYED\_COUNTER\_TO\_ELAPSED\_TIME.get(this);

}

public boolean isFakeEngagement() {

return FAKE\_ENGAGEMENT\_FEATURES.contains(this);

}

public boolean isBlinkEngagement() {

return BLINK\_ENGAGEMENT\_FEATURES.contains(this);

}

/\*\*

\* Check if an increment is eligible for emitting

\*/

public static class IncrementChecker {

private final IntNormalizer normalizer;

public IncrementChecker(IntNormalizer normalizer) {

this.normalizer = normalizer;

}

IncrementChecker(TweetFeatureType type) {

this(type.getNormalizer());

}

/\*\*

\* Check if a value change is eligible for output

\*/

public boolean eligibleForEmit(int oldValue, int newValue) {

return normalizer.normalize(oldValue) != normalizer.normalize(newValue);

}

}

}